실시간 에이전트들을 위한 혼합형 계층 구조

김하빈°, 권기덕, 김인철
경기대학교 정보과학부
{talkable°, kdkwon, kic}@kyonggi.ac.kr

A Hybrid Hierarchical Architecture for Real-time Agents

Ha-Bin Kim°, Ki-Duck Kwon, In-Cheol Kim
Dept. of Computer Science, Kyonggi University

요 약

기존의 실시간 에이전트 환경에서는 에이전트 구조에서 고려하지 않았던 높은 복잡성의 문제를 해결하기에 환경에 대한 고려가 부족하여 구현 시 충분한 지침으로 삼기에는 부족하거나 적합하지 않았다. 본 논문에서는 이러한 고려하여야 할 환경에서 필요한 요소들을 기존의 계층기반 에이전트 구조를 보완한 혼합형 구조를 이용하여 행위기반 구조를 설계하고 구현하였다. 분산적이며 실시간으로 동작하는 환경에서는 효율적이고 범용적으로 사용 할 수 있는 행위 기반 구조가 요구 된다. 본 논문에서 제시하는 에이전트 구조는 행위의 논리적 상하계층에 중점을 둔 계층별 분류를 사용하지 않고, 범주 분류한 RtABCM을 사용하여 복잡한 실시간 환경에 유연하게 적응할 수 있는 구조를 제안하였다. 이를 통하여 계층의 단계와 병렬적으로 진행이 가능한 동일한 계층 행위의 수에 제약을 두지 않게 되어 정적인 계층 구조에서 오는 제약의 한계를 극복하고 있다. 또한 행위의 객체화와 이를 위한 구성 요소의 지원으로 실시간 환경에 대한 다중의 행위나 계획 진행에 대한 유연한 진행, 양방향성을 지원하는 확장된 행위모델, 설계와 구현에 있어 자유롭고 유연한 모델을 제시하고 있다. 본 논문에서는 RtABCM에 적용한 행위기반 구조를 실시간 에이전트 환경인 GameBots에 적용시켜 구조의 실시간 환경에 대한 적응성을 증명하고 있다.

1. 서론

고전적인 에이전트에 대한 연구는 비교적 정적이고 단순한 환경에서 행해졌으며 이러한 환경에 적합한 에이전트의 구조로 제시된 범용의 에이전트 구조들은 지속적으로 그 복잡성이 증가하고 있는 에이전트 환경에 유연하게 적용 하기에는 미흡하다. 또한 행위의 복잡성이 증가함에 따라 단 방향으로만 진행되고 제한적이며 단순한 기능 구조만으로 구성된 기존 구조를 이용한 설계는 행위 별 구현의 복잡성을 증가시키게 된다. 따라서 이러한 문제를 해결 하기 위한 현대적 에이전트 환경에 유연하게 동작하는 에이전트 행위 기반 구조의 제시가 요구 되고 있다. 이러한 복잡성 높은 실시간 환경의 예로 게임 환경을 들 수 있다. 최근의 게임은 3차원 환경에서 복잡한 물리 법칙이 적용되어 직접 제어해 주어야 할 부분이 많아졌으며, 빠르게 환경이 변화 한다. 최근에는 더욱 사실적인 에이전트의 연구를 위해 이러한 게임환경을 이용하는 예가 늘고 있다.

2. 관련 연구

범용의 에이전트의 구조는 숙고형 구조나 반응형 구조, 이 두 가지를 결합한 혼합형 구조로 나누어 볼 수 있다. 이 혼합형 구조에는 복잡한 추론과정 없이 환경변화에 바로 반응할 수 있는 반응형 모듈과 실세계에 대한 기호모델을 바탕으로 계획을 세우거나 의사결정을 내리는 숙고형 모듈을 함께 내포하며, 숙고형 모듈에 비해 반응형 모듈에 더 우선권(precedence)을 부여하여 중요한 환경변화에 빠른 반응을 제공할 수 있도록 하였다. 1987년 Georgeff와 Lansky에 의해 개발된 PRS(Procedural Reasoning System)[2]은 대표적인 BDI 에이전트 구조이다. 이 구조는 내부에 기호로 명시된 믿음(belief), 소망(desire), 그리고 의도(intention)들의 집합과 계획 라이브러리(plan library)를 포함하고 있다. PRS는 실행 중인 지식영역들에 대해 명시된 바대로 문맥이 지켜지고 있는 지를 수시로 점검하여 문맥이 만족되지 않으면 곧바로 해당 지식영역의 실행을 중단함으로써 환경변화에 신속히 반응할 수 있다. 1992년 Ferguson이 발표한 TouringMachine[3]의 구조는 환경과 직접 맞닿는 인지부(perception)와 행동부(action), 그리고 하나의 제어 틀(control framework)속에 포함되어 있는 세 개의 제어 계층들로 구성되어 있다. 각 제어 계층은 에이전트의 행동을 결정하기 위해 독립적으로 병행 수행되는 하나의 프로세스이며, 제어 틀은 이들 계층들간의 중재 역할을 수행한다. 세 계층들은 모두 하나의 제어 틀(control framework) 내에 포함되어 있으면서 메시지 교환을 통해 서로 교신할 수 있다. 이러한 제어 틀의 목적은 계층간의 중재 역할을 하는 것이며 특히 제어 규칙(control rule)들을 이용하여 서로 다른 계층들에서 제안하는 동작들간의 충돌을 처리하는 것이다. Muller에 의해 제안된 InteRRaP[7]은 계층구조이며, 연속적으로 놓여진 각 계층은 하위 계층보다 더 높은 추상화를 나타낸다. 이 시스템 행위패턴들의 실행은 에이전트의 궁극적인 목표 달성을 위해 상위 계층들인 행위-기반 제어부와 협동 제어부로 하여금 단일 에이전트 계획 또는 협동계획을 생성하도록 요청할 수 있다. 이와 같은 과정을 거쳐 최종적으로 다시 실세계 인터페이스부에 의해 기본 동작들을 실행하거나 메시지들을 전송하게 된다. 앞서 열거한 에이전트 구조들은 나름대로 더욱 복잡성 높은 문제를 처리하기 위하여 흐름이나 계층별로 분리하여 각 모듈의 설계를 경량화 하며, 행위 결정부의 설계에 대한 방법을 제시하고 있다. 특히 InteRRaP을 비롯한 몇몇 계층형 구조에서는 에이전트 행위 제어 모듈 내부의 계층화 방법을 사용하여 더욱 복잡한 행위 구현에 대해 적극적인 해법을 제시하고 있다. 하지만 과거에 이러한 행위구조는 비교적 단순하며 정적인 환경에서 실험이 이루어 졌고 실시간이며 복잡한 진행이 요구되는 행위에 대한 대안을 적극적으로 제시 하지 못하였다.

3. 에이전트 환경

하드웨어가 발전하며 에이전트의 활용 범위가 넓어짐에 따라 에이전트 환경은 변화하고 있다. 변화하는 에이전트의 환경이 가지는 특징은 다음과 같다.

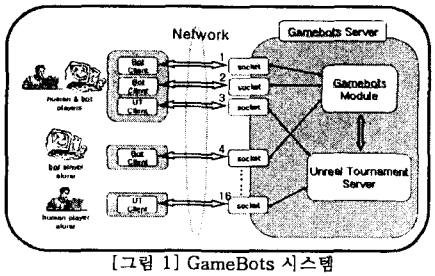
- **실시간** : 급격하게 변하는 환경에서는 에이전트가 사고하고 행동을 취할 때까지 기다려 주지 않는다. 때문에 실시간 에이전트는 어느 시간에 행동을 수행 할 것인지 결정 할 수 있어야 하며 숙고가 필요한 판단과 반응적 판단의 혼합형 구조가 요구된다.
- **다중 에이전트** : 환경이 다른 에이전트들에 의해 변화할 수

있다. 따라서 에이전트로 하여금 행위의 진행에 대한 예측을 더욱 어렵게 만든다.

- **환경의 복잡성** : 에이전트 환경이 더욱 사실적으로 바뀜에 따라 에이전트 행위 또한 증가하였다. 이러한 행위들은 장/단기의 단순한 분류만이 아니라 병렬적으로 동작 할 수 있는 행위가 존재하기도 하며 때로는 행위의 관계가 종속적 관계를 유지하지 않을 수 있다.

이러한 에이전트의 환경적 특징들이 본 논문의 환경에는 다음과 같은 특징들을 가지게 하였다..

3차원 환경: 실수 수준의 3차원 물체로 이루어진 환경
사실적 환경: 물리법칙과 잡음이 적용되는 사실적인 환경
제한적 인지능력: 환경정보에 대한 정보 습득의 제한
실시간: 동적 행위 진행으로 인한 환경의 지속적인 변화
다중에이전트: 환경을 변화 시킬 수 있는 요소
메시지통신: 동기, 비동기 기반의 메시지를 통한 에이전트간 통신으로 인한 정보의 다양성
행위의 다양화: 에이전트가 행할 수 있는 행위의 유형 증가

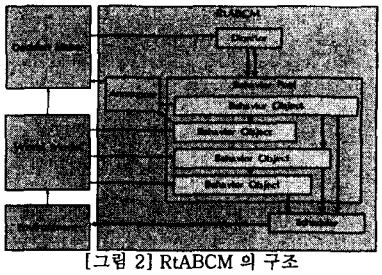


[그림 1] GameBots 시스템

[그림 1]은 본 논문에서 실험 환경으로 사용하는 GameBots 시스템의 구조이다. GameBots은 견고한 3D환경을 제공하는 Epic사의 Unreal Tournament 게임 서버와 연동하며 보트 클라이언트들은 GameBots를 통하여 게임 환경에 접근 하며 인간 플레이어와 에이전트가 동일한 환경을 공유한다. 이때 에이전트가 인지할 수 있는 정보는 인간 플레이어의 인지 범위를 초과하지 않아 제한적 인지능력 특성요소를 보인다. 또한 복잡한 3차원 환경과 물리 법칙의 적용하여 3차원 환경특성과 사실적 환경 특성요소를 만족하며 1/10초 단위로 인지하며, 수행하여야 하는 실시간 요소를 포함하고 있다. 멀티에이전트 요소로는 최대 15개의 서로 협업하거나 경쟁할 수 있는 다른 게임 클라이언트들이 존재하고 이들은 각기 환경을 끊임없이 변화시키며 전투나 충돌을 통해 직접적으로 다른 플레이어에 영향을 주기도 하여 서로간의 행위 예측을 더욱 어렵게 하고 있다. 각 게임 클라이언트는 이동하고, 전투하고, 대화하는 20가지 이상의 기본 행위를 가지고 있으며 각 행위는 단발성 행위이거나, 한번 실행한 행위가 오랫동안 지속될 수 있는 특성을 지니고 있다. 특정한 행위는 그 행위가 진행되는 동안 다른 행위를 발생 시킬 경우 먼저 진행되고 있던 행위가 중단되지 않을 수 있는 등 다양한 특성의 행위들이 존재한다.

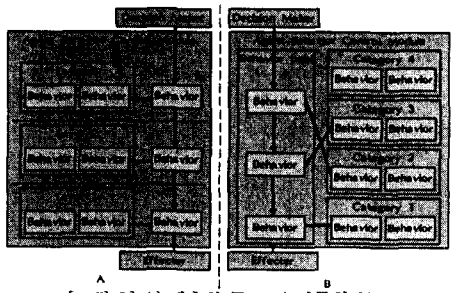
4. RtABCM 구조

2절에서 기술한 환경의 특징으로 인해 기존에 소개되었던 에이전트 구조만으로는 에이전트 구조의 충분한 지침이 되지 못하였다. 분산적이며 실시간으로 동작하는 환경에 대한 모든 고려 사항을 상위 계층의 행위 결정부에서 구현하는 것은 어려운 일이다. 때문에 이러한 환경에서 효율적이고 범용적으로 사용 할 수 있는 행위 기반 구조가 요구 되는데 본 논문에서는 RtABCM(Real-time Agent Behavior Control Module)을 사용하여 복잡한 실시간 환경에 유연하게 적응할 수 있는 구조를 구현하였다. [그림 2]는 RtABCM의 구조를 나타내고 있다. RtABCM은 행위객체(behavior object)를 담고 있는 행위 풀(behavior pool)과 행위객체의 제어를 위한 모듈로 구성된다. Director는 행위객체의 생성, 교체 그리고 삭제를 하며 Announcer는 의사 결정부에게 행위객체에서 발생하는 이벤트를 보고한다. 그리고 Scheduler는 최종적으로 선택된 행위 중 수행 가능한 행위를 Effecter에 전달하게 된다.


[그림 2] RtABCM 의 구조

RtABCM은 기존 계층 구조와는 달리 설계 단계부터 제한된 계층을 두지 않고 계층 구조 대신 범주(category) 구조를 사용하고 있다. 범주 구조란 행위를 계층 종속 관계가 엄격하지 않은 여러 개의 범주로 나누고 실행 시간 동안 행위의 선택에 따라 동적인 계층이 형성되는 구조이다. 본 구조는 기존의 계층구조와 전혀 다른 구조가 아닌 계층을 동적으로 연결하는 구조라 표현 할 수 있다.

4.1 범주를 사용한 행위 계층 구조


[그림 3] A) 계층형 구조 B) 범주형 구조

[그림 3]은 계층형 구조와 범주형 구조를 비교하고 있다. 범주형 구조는 계층형 구조와는 달리 행위의 논리적 상하 관계에 따른 분류가 아닌 행위의 특성에 따라 다른 행위범주로 분류한다. 실세계에서 인간이 움직일 경우 목표 지점으로 움직이는 단순한 행위조차도 많은 사고를 하게 된다. 목표를 정하고, 그곳까지 이동하는 동안 제대로 가고 있는지, 혹은 장애물이 있는지를 검사하고 장애물이 있다면 목표까지 이동하는 계획 자체에 큰 고려를 하지 않고 피해가거나, 그 장애물이 목표 달성에 영향을 줄 정도라면 계획을 바꿔야 한다는 ' 무의식적 사고' 를 하게 된다. 더 낮은 계층 수준의 행위로는 다리를 움직이며, 또한 무릎의 각도의 변화까지 감지하고 있다. 이 때 논리적 계층이 비슷하거나 차이가 나는 여러 행위를 병렬적으로 진행 하게 되며, 행위에 대한 진행 사항을 감시하게 된다. 이와 같은 복잡한 사고의 진행은 기존의 정형화된 계층 구조에 적용하기 위해서는 계층마다 구현 내용이 너무 복잡해 지게 된다. RtABCM에서는 행위를 계층별로 분류하지 않고, 범주별 분류를 하여 계층의 단계와 병렬적 진행이 가능한 같은 계층 행위의 수에 제약을 두지 않음으로써 정적인 계층 구조에서 오는 제약의 한계를 극복하고 있다. 또한 RtABCM에서 각각의 행위 객체들은 스스로 행위에 대한 진행의 평가를 하게 되며, 중대한 사항일 경우 상위 행위객체로 보고하여 ' 무의식적 사고' 를 하는 형태를 지닌다. 이로 인해 행위는 단지 반응의 범위를 벗어나 행위의 상태를 포함하여 기존 계층구조에서의 한 개의 계층 역할을 하게 되어 복잡한 환경에 대한 상위 행위 결정부와 각 행위별 구현 단위를 경량화 할 수 있다.

4.2 행위객체

인간은 일반적으로 어떠한 작업에 대한 판단을 내릴 경우, 각 단위의 행위에 대한 직접적이고 구체적인 계획을 세우기 보다

개념적이고 포괄적인 계획을 세운 후 계획을 수행하면서 환경에 적응해서 세세한 행동을 결정하게 된다. 이와 같이 RtABCM은 판단부에서 수행에 영향을 미치는 환경적 요인에 대해 많은 고려를 하지 않더라도 스스로 환경에 대처하는 모듈을 목표로 설계된 것이다. RtABCM은 실시간이며 복잡한 판단이 요구되는 상황에 대하여 행위기반 구조로 유연하게 동작하는 병렬적 행위객체를 설계하고 필요한 구성 요소를 제시하여 그를 위한 해결책을 제시하고 있다. 모든 행위는 객체로 표현되며, 행위객체는 자신의 범주, 중요도, 다른 행위와의 연관성, 행위 제약을 표시하는 속성을 가지고 있다. 해당 행위를 진행하는 동안 상태를 평가하고 스스로 교정하거나 보고 할 수 있는 함수를 포함한다. 실행 시간 동안 작동중인 행위객체는 기존의 계층 구조에서 한 계층의 역할을 담당하게 되므로 이외의 행위 계층이 가져야 하는 속성들을 도메인에 맞도록 추가적으로 포함할 수 있다. 행위풀은 행위객체를 저장한다. 저장된 행위는 객체화 되어 행위가 선택된 순간부터 실행되거나 다른 모듈에 의해서 삭제 될 때까지 행위 범주별로 저장 되는데, 각 범주별로 행위풀에서 선택된 행위들은 대기열에 위치하게 되며, 그 구동이 scheduler에 의해 허가 받게 되면 행위는 실행열로 이동한다.

4.3 RtABCM 구성 요소

병렬적이고 유동적이며 대화형 행위객체의 구동을 위해 RtABCM은 다음과 같은 구성요소를 포함하고 있다.

-Director

*Director*는 선택된 행위를 객체형태로 변환, 행위 대기 열에 추가 시키는 역할을 한다. 매번 행위객체를 추가 할 때 director는 이미 대기 열에 위치한 행위객체와 새로이 선택된 행위객체를 비교하게 되는데, 그 관계를 고려하여 행위객체의 추가, 교체, 삭제행위가 발생하게 된다. 기본적으로, 동일한 범주의 행위객체, 즉 동시에 수행 할 수 없는 행위객체는 동시에 행위 풀에 저장 될 수 없다.

-Announcer

Announcer는 행위객체의 이벤트를 상위 행위 결정부에 전달하는 역할을 한다. 상위 행위 결정부에서 직접적으로 발생하지 않은 행위객체의 이벤트는 RtABCM내부의 해당 행위객체를 발생시킨 행위객체에게만 전달 되므로 최상위 행위 결정부는 스스로 발생시킨 행위객체의 이벤트 처리만 고려하는 것이 가능하다.

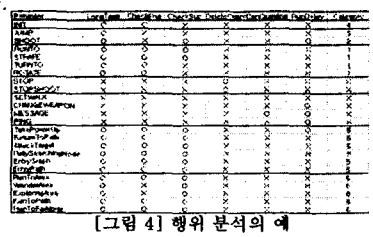
-Scheduler

대기 열에 있던 행위 객체들을 등급, 시간 제어 속성들을 고려하여 선택, 실행 열로 옮기게 된다. 계층별로 한 시간 단위당 한 개의 행위만을 수행 하게 되지만, 행위가 실행된 후 완료되기 전까지 감시 할 수 있도록 기억하게 된다. 실행 열에 있는 행위 객체들은 환경에서 제공하는 행동 시간 단위, 행위별 재실행 시간을 고려하여 수행하게 된다. 행위의 수행은 직접적인 환경에의 작용뿐 아니라 새로운 하부 계층 행위객체를 만들어 내기도 한다.

5. 구현

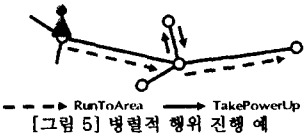
본 연구에서는 *GameBots* 환경에서 동작하는 KGBot을 구현하였고, RtABCM을 행위 제어 모듈로 사용, 그 성능을 실험하였다. GameBots환경에서 KGBot는 다수의 게임 클라이언트가 활동하는 환경에서 적과의 전투, 아이템 습득, 지점의 점령 등의 논리적인 병렬적 수행이 가능한 다양한 성격의 행위를 한다.

KGBot 행위제어모듈의 구현에서 각 행위 객체의 설계 및 구현이 중요한 사항으로 인식 되었고 그 행위의 치밀한 분석이 요구 되었다.



[그림 4] 행위 분석의 예

[그림 4]는 KGBot의 행위 분석에 사용한 예이다. 이 과정에서 API에서 제공하는 직접적인 효과를 주는 행위 중 논리적 행위, 즉 다른 행위를 발생시키거나 내부적 사용만을 위한 행위를 분류하였다. 이러한 분석 결과를 통하여 해당 행위의 구현에 대한 *개략적인 설계*와 *행위의 범주 정의*에 직접적인 *지침*이 될 수 있었다.



[그림 5] 병렬적 행위 진행 예

[그림 5]는 병렬적 행위 진행의 예를 설명한 그림이다. 그림에서 에이전트는 ' 영역으로 이동' 행위의 진행 도중 게임 내에서 도움이 될 수 있는 아이템을 발견하였다. 이때 아이템 습득을 위한 행위를 시도 하는데, 이 행위는 ' 영역으로 이동' 행위와는 다른 범주의 행위로, ' 영역으로 이동' 행위를 중단하지 않게 된다. 하지만 두 행위는 모두 단순 이동 행위를 발생시키게 된다. 두 개 이상의 단순 이동 행위는 같은 범주의 행위이기 때문에 매 사이클마다 두 행위 중 한가지 만이 선택되게 되는데, 아이템 습득 행위에서 발생한 단순 이동 행위가 ' 영역으로 이동' 행위보다 더 높은 중요도 속성을 가지게 되기 때문에 이 행위를 우선적으로 선택하고 ' 영역으로 이동' 행위를 유보시키고, 수행한 행위가 끝나고 나면 ' 영역으로 이동' 행위를 지속적으로 수행하게 된다. 이러한 방식은 계획을 수립하고 그에 따른 행동을 전개할 때 성격이 다른 행위를 논리적으로는 병렬적인 진행이 가능하게 한다. KGBot에서는 이를 통하여 지역을 점령하는 도중 필요한 다른 행위들 즉, 적과의 전투 또는 아이템 습득 행위를 지역의 점령 행위에 논리적인 영향을 주지 않고 병렬적으로 진행 할 수 있었다.

6. 결론

본 논문에서는 에이전트 환경에 대한 RtABCM의 사용, 특히 동적 계층이라 표현 될 수 있는 범주를 사용한 방법으로 복잡한 실시간 환경에서 기존의 에이전트 구조 중 빈번히 사용되는 계층형 구조를 확장한 형태의 행위 객체를 사용한 설계 방법을 제시하고 이러한 환경에서 유연한 수행을 만족하기 위한 행위 객체를 위한 모듈 구성 요소를 소개 하였다. 행위를 객체단위로 설계하여 복잡한 행위에 대한 강건성을 높이고, 유연한 행위 구현으로 상위 행위 결정부의 설계에 자유로운 병렬적 행위 진행을 선택 할 수 있었다. 본 논문에서 소개하고 있는 구조는 RtABCM을 사용한 현대적인 복잡성 높은 실시간 환경에서 범용적으로 사용 할 수 있는 에이전트 구조 설계의 지침을 목표로 하였다.

참고문헌

[1] Adobbati, R., Marshall, A.N., Scholer, A., Tejada, S., Kaminka, G.A., Schaffer, S., Sollitto, C. GameBots: A 3D virtual world test bed for multiagent research. In *Proceedings of the second International Workshop on Infrastructure for agents, MAS, and Scable MAS*, 2001.

[2] Ferguson, I. A. TouringMachines: An Architecture for Dynamic, Rational, Mobile Agents. PhD thesis, Clare Hall, University of Cambridge, UK. 1992.

[3] Georgeff, M. P. and Lansky, A. L. Reactive reasoning and planning. In Proceedings of AAAI-97, pages 677-682, 1987

[4] Jose M. Vidal, Paul Buhler. Teaching Multiagent Systems using RoboCup and Biter. Interactive Multimedia Electronic Journal of Computer-Enhanced Learning, 4(2), 2002.

[5] Klaus Dorer. Extended Behavior Networks for the magmaFreiburg Team. RoboCup-99 Team Descriptions, Simulation League, Team magmaFriburg, pages 79-83. 1999.

[6] Muller, J. P., Pischel, M., and Thiel, M. A pragmatic approach to modelling autonomous interacting systems. In Wooldridge, M. and Jennings, N. R., editors, Proceedings of the 1994 Workshop on Agent Theories, Architectures, and Languages, pages 226-240, Amsterdam, The Netherlands. 1994